



*Petaflops-systems Operations
Working Review*



Thomas Sterling
California Institute of Technology
NASA Jet Propulsion Laboratory
August 13, 1998

POWR Workshop Overview



- Petaflops initiative & context
- Objectives
- Charter & Guidelines
- 3 Pflops system classes
 - COTS clusters
 - MPP system architecture
 - Hybrid-technology custom architecture
- Specific group results
 - Summary findings
 - Open issues
 - Recommendations
 - Conclusions

TOP500

(June 18, 1998)

We try to implement links to the WWW-Homepages of all the sites listed in the table. Please send address of Homepages to top500@rz.uni-mannheim.de.

A table of 500 entries (ca. 120 KB) is loaded now ...

Rank	Manufacturer	Computer	Rmax	Installation Site	Country	Year	Area of Installation	# Proc	Rpeak	Nmax	N1/2
1	Intel	ASCI Red	1338000	Sandia National Labs Albuquerque	USA	1997	Research	9152	1830400	235000	63000
2	S6I	T3E1200 LC1080-512	891500	Government	USA	1998	Classified	1080	1296000	259200	26400
3	S6I	T3E900 LC1248-128	634200	Government	USA	1997	Classified	1248	1123200	.	.
4	S6I	T3E900 LC840-128	450500	United Kingdom Meteorological Office Bracknell	UK	1997	Research Weather	840	756000	.	.
5	S6I	T3E LC1024-128	448600	NASA/Goddard Space Flight Center Greenbelt	USA	1998	Research Weather	1024	614400	119808	19008
6	Hitachi/Tsukuba	CP-PACS/2048	368200	Center for Computational Physics, Univ of Tsukuba Tsukuba	Japan	1996	Academic	2048	614000	103680	30720
7	S6I	T3E LC784-128	342800	Max-Planck-Gesellschaft MPI/IPP Garching	Germany	1997	Research	784	470400	104832	17280
		T3E900									



Comparsion to Present Technology

<u>IMPROVEMENT</u>	<u>TODAY</u>	<u>Pflops</u>	
PERFORMANCE 1000	1 Teraflops	1000 Teraflops	X
POWER	2 Mflops/watt	1000 Mflops/watt	X 500
COST	\$250/Mflops	\$0.25/Mflops	X 1000
EFFICIENCY	10%	50%	X 5

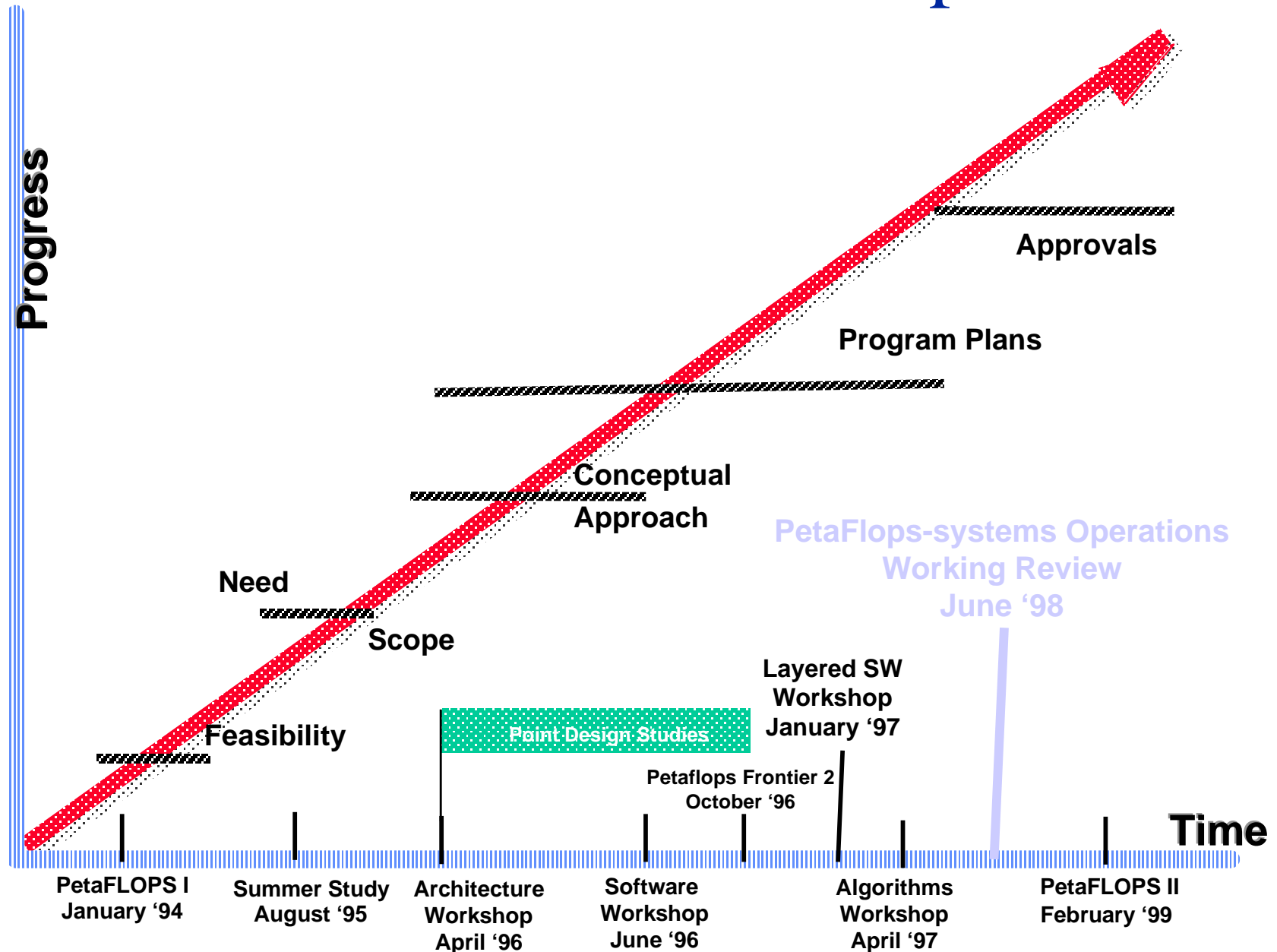
FLOORSPACE

1600 sq ft/Tflop

1 sq ft/Tflop

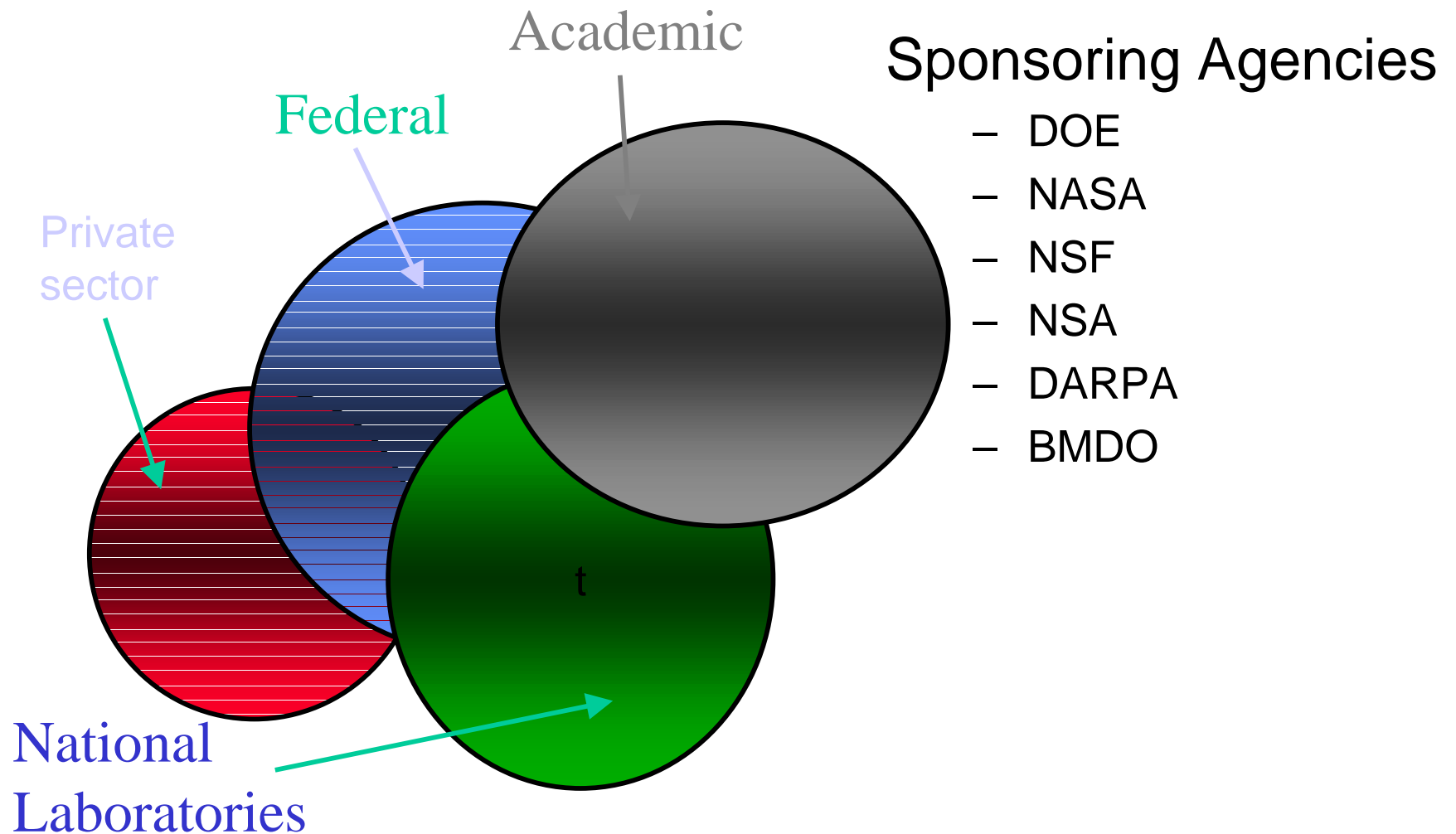
X 1600

PetaFLOPS Workshops



Workshop series... background:

Community & Sponsoring Agencies



Previous Accomplishments

- Recognizing the issue
 - Important applications identified requiring Pflops machines
 - Industry is considering 50-100 Tflops systems
 - Interests by US and other governments
- Hardware/Architecture Implementation approaches
 - Driving factors (e.g. latency, parallelism, bandwidth)
 - Architecture archetypes
 - through NSF Point Design studies
 - distributed clusters of cots systems
 - new MPP structures of commercial components
 - innovative architectures incorporating advanced technologies
 - special purpose devices
- Software and algorithm requirements

Uncompleted Tasks

- Programming models
- Operating systems
- Scaling models for applications and algorithms
- Definitive Analysis of Limits of Moore's Law for CMOS
- Compelling Applications with Watertight Justification of Need for Petaflops and Analysis of Configuration Parameters
- Characterization and analysis of architecture efficiency under real-world workloads

POWR Goals

- Fulfill the promise of the Petaflops Initiative
- Provide the basis for future sponsored research programs
- Answer the questions of What and How
 - destinations and the paths
- Ground work for final culminating community forum
 - 2nd Workshop on Enabling Technologies for Peta(fl)ops Computing
 - February, 1999

POWR Objectives

- Consider range of alternative system approaches
 - COTS Clusters
 - commercial only node and system architecture
 - MPPs
 - mostly commercial parts with custom system architecture
 - HTMT
 - custom technologies and parts with custom node and system architecture
- Description of systems and their comprising functional elements
- Identification of unresolved issues
- Definition of tasks to be performed
- Develop a succinct report for Pflops-2 workshop and research program managers

Approach

- Vertically integrated systems approach
 - keystroke to instruction-issue
 - applications algorithms, semantics, compile time, runtime, architecture support
- Pursue three most promising systems concepts
- Convene working groups for each system class
- Emphasize performance driven factors
- Block diagram for each system
- Functional and interface description
- Open issues

The *High Cs* to crossing to Petaflops Computing

- Capability
 - Computation rate
 - Capacity of storage
- Cost
 - Component count
 - Connection complexity
- Consumption of power
- Concurrency
- Cycles of latency
- Customers
- Confidence

Strategic Questions

- What is the performance responsibility for :
 - hardware architecture
 - runtime software
 - compiler analysis
 - algorithm & language
- What performance related information needs to flow between successive system levels?

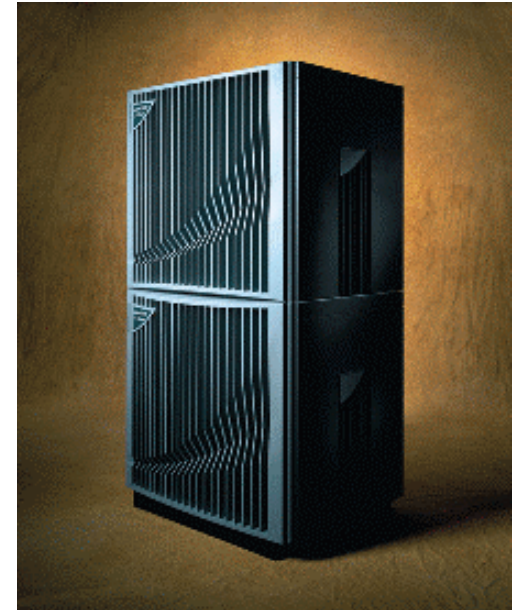
Invited Plenary Talks

- Keynote Address: “From Here to Petaflops”, B. Smith
- “MPP Architecture for Petaflops”, T. Agarwala
- “The First Steps towards Petaflops”, P.H. Smith
- “Impressions of Accomplishments to Date”, P. Messina
- “HTMT Architecture for Petaflops”. T. Sterling
- “PIM Technology and Architecture”, P. Kogge
- “Challenges and Tasks to Achieving Petaflops”, V. Kumar, D. Keyes, J. Torrellas, J. McGraw
- “Hierarchical Granularity MPP Architecture”, J. Fortes
- “Making COTS Petaflops Work”, M. Warren
- “MPP Emulation Requirements for Algorithm Design”, P. Woodward



MPP Petaflops System

- COTS chips and industry standard interfaces
- Custom glue-logic ASICs and SAN
- New systems architecture
- Distributed shared memory and cache based latency management
- Algorithm/application methodologies
- Specialized compile time and runtime software



Summary of MPP

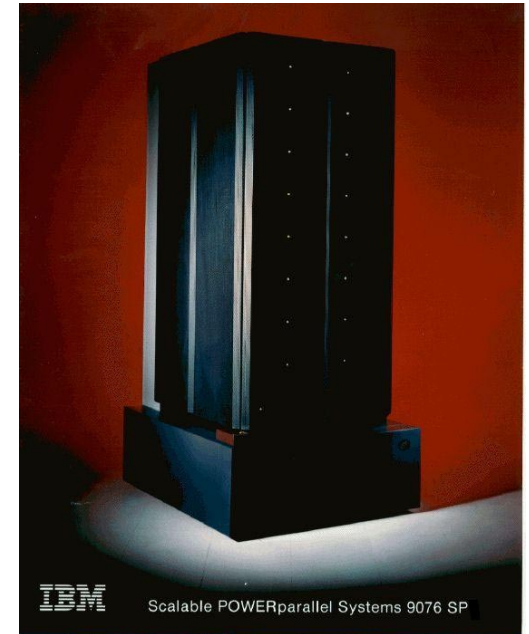
- processor: 3 GHz, 10 Gflops
- # processors: 100,000
- memory: 32 Tbytes, DRAM, 40ns access time local
- interconnect: frames switched, 128 Gbps/channel
- secondary storage: 1 Pbyte, 1 ms access time
- distributed shared memory
- latency management: cache coherence protocol





COTS Clustered Petaflops System

- NO specialized hardware
- Leverages mass market economy of scale
- Distributed memory model with message passing
- Incorporates desktop/server mainstream component systems
- Integrated by means of COTS networking technology
- Augmented by new application algorithm methodologies and system software



[illegible]

*Winner
Performance*

Michael S. Warren
Alamos National Laboratory

John K. Salmon
California Institute of Technology

*Winner
Price/Performance*


S. Warren and M. Patrick Goda
Alamos National Laboratory

Donald J. Becker
NASA Goddard Space Flight Center

C. Salmon and Thomas Sterling
California Institute of Technology

Grégoire S. Winckelmans
Catholic University of Louvain

... of their superior effort in practical parallel-processing research


Ed Parrish, Editor-in-Chief, Computer

Presented by **COMPUTER**

ates 'Beowulf' computers: Right
line price, speed for schools, labs

By John Yankey
Gannett News Service

WASHINGTON Supercomputers were once the elegant, untouchable Formula One racing cars of computing. They emerged from sterile laboratories as sleek, futuristic space probes, and cost almost as much.

Now they are being assembled in academic garages from off-the-shelf parts at a fraction of the cost.

There is a great future for this, said Michel Valletier, head of the department of physics and atmospheric science at Drexel University, which built one of the first-generation Borealis machines.

"You can build with whatever parts and crystals you want," Valletier didn't have much space, so the idea was very attractive.

He's counting on the fact that most supercomputers typically consist of several dozen boards, each of them packed with several dozen personal computers wired together so all the processors, or computational engines can talk to each other as they work together.

They were made possible simply by NASA, which developed the networking software to make them virtually all the same, and by the fact that they will compute

These new Beowulf computers, named for the ancient English hero, are emerging from the garages of America in the 1990s.

For thousands — not millions — of dollars, scientists can obtain the computer power and model and study everything

that makes it possible for smaller, less wealthy universities and laboratories to obtain world-class computing power.

Beowulf computers costing from \$30,000-\$50,000 have already broken the gigaflop barrier (a gigaflop is 1 billion floating-point calculations per second). Several

The Beowulf computing philosophy is to divide and conquer: Beowulf computers break down a task into smaller pieces that can be distributed among the individual processors in the computers, where they are dealt with simultaneously.

See 11/21/93, Free-Line page 10

Newsletter has guidelines that say medical bills an employee suffers playing on a company sports team shouldn't come to the company.

* John Glisch, special projects editor, 242-3968, 9 a.m.

Piles of PCs for one solution

The new Beowulf class computers consist of dozens of standard personal computers wired together. They cost tens of thousands of dollars but deliver million-dollar computing speeds of half a billion calculations per second or more. Beowulf computers

1 A problem needing computer analysis is broken into a series of sub-problems by software, then delegated to dozens of Pentium-powered computers that can communicate with

Problem

2 The processors work through the individual segments of the problem, processing the pieces simultaneously.

3 Instead of a problem...

Produced by the *Journal of the American Veterinary Medical Association* and the *Journal of the American College of Veterinary Surgeons*

INTERNET WEEK

BY JEFF CARUSO

Though these so-called Beowulf-class machines have been in research

ETHERNET, PAGE 53 ➤

BY TIM WILSON
Washington, D.C.

It could happen if officials at the General Services Administration can garner support for an ambitious

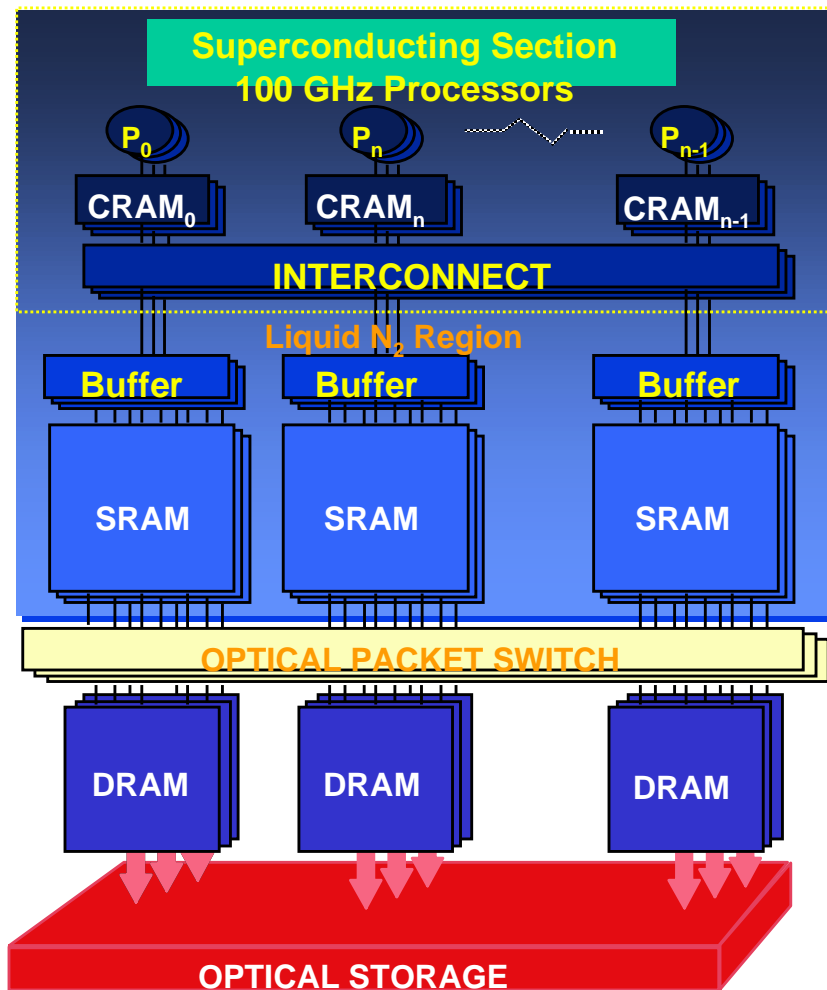
partial
ing—
transac
on bel
compu
custom
The
aimed
otherv
reseller
aggress
multip
utors f
price,
gram f
to conc
ternet
Cou
initiati
to the

				(Canada)			Database				
313	IBM	SP2/104	19340	MCI	USA	1994	Industry Telecomm	104	27620	.	.
314	IBM	SP2/104	19340	NIH (National Institutes of Health) Bethesda	USA	1997	Research	104	27620	.	.
315	Digital	Avalon Cluster	19330	Los Alamos National Laboratory / CNLS Los Alamos	USA	1998	Academic	68	72480	30464	14376
316	Fujitsu	VPP300/9	19225	ECMWF Reading	UK	1997	Research Weather	9	19800	.	.
317	Sun	HPC 10000	18670	Ameritech	USA	1998	Industry Telecomm	44	22000	19968	2496
318	Sun	HPC 10000	18670	Worldcom	USA	1998	Industry Telecomm	44	22000	19968	2496
319	Fujitsu	VPP300/BE	18600	Kansai University	Japan	1997	Academic	8	19200	41600	2400
320	Fujitsu	VPP300/BE	18600	Osaka Gas., Ltd Osaka	Japan	1998	Industry Chemistry	8	19200	41600	2400

Summary of COTS Cluster

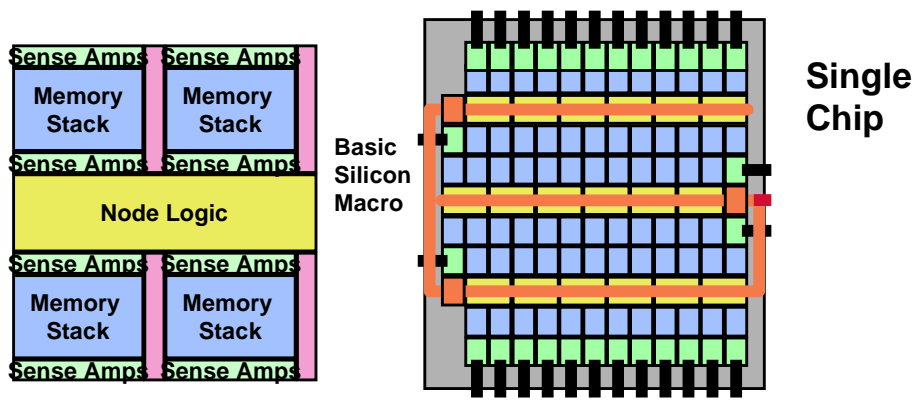
- processor: 3 GHz, 10 Gflops
- # processors: 100,000
- memory: 32 Tbytes, DRAM, 40ns access time
- interconnect: degree 12 n-cube, 20 Gbps/channel
- secondary storage: 1 Pbyte, 1 ms access time
- distributed memory, 3 level cache, 1 level DRAM
- latency management: software

Hybrid Technology Petaflops System

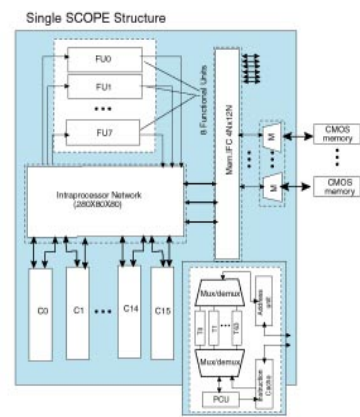
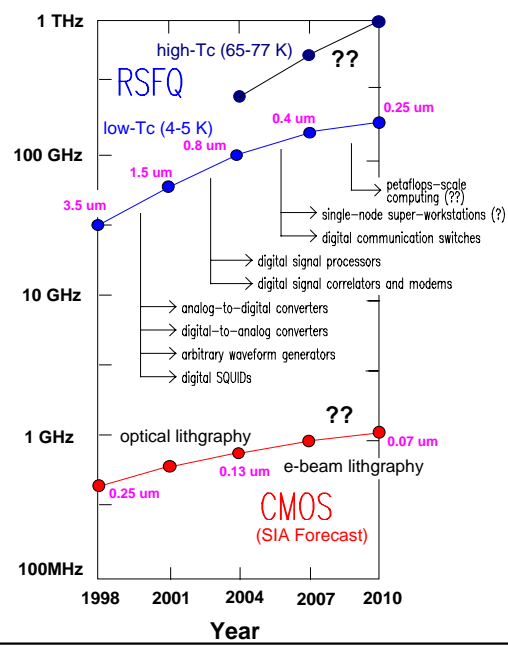


- New device technologies
- New component designs
- New subsystem architecture
- New system architecture
- New latency management paradigm and mechanisms
- New algorithms/applications
- New compile time and runtime software

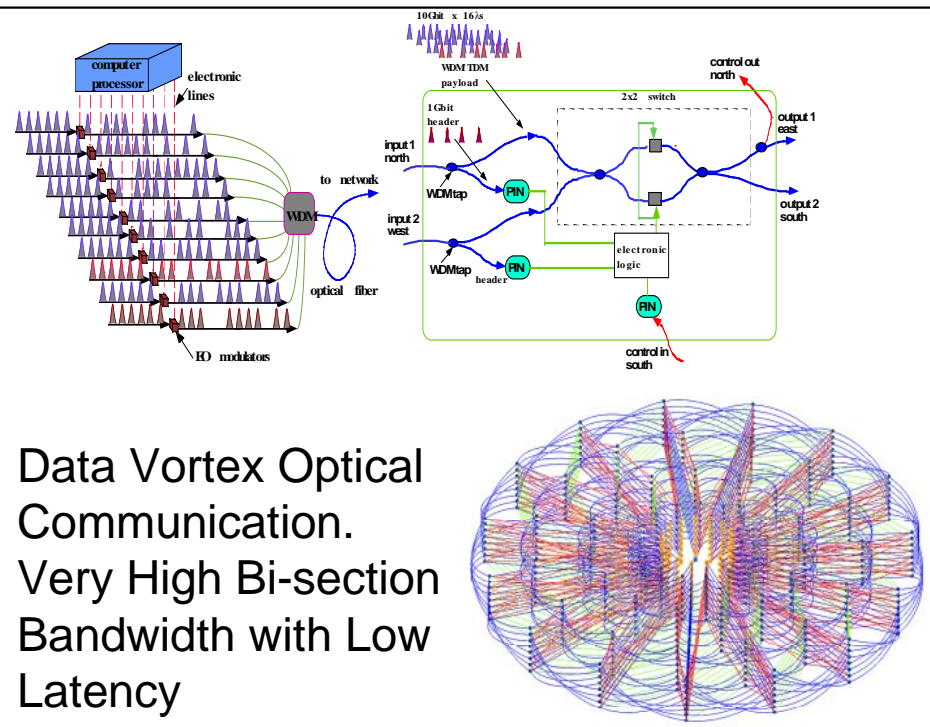
Complementing Technologies Yield Superior Power/Price/Performance



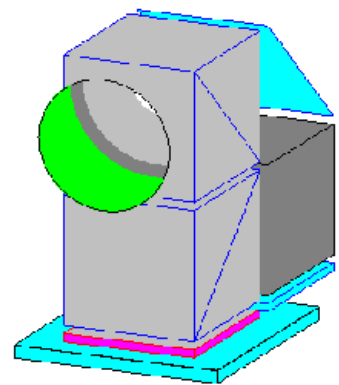
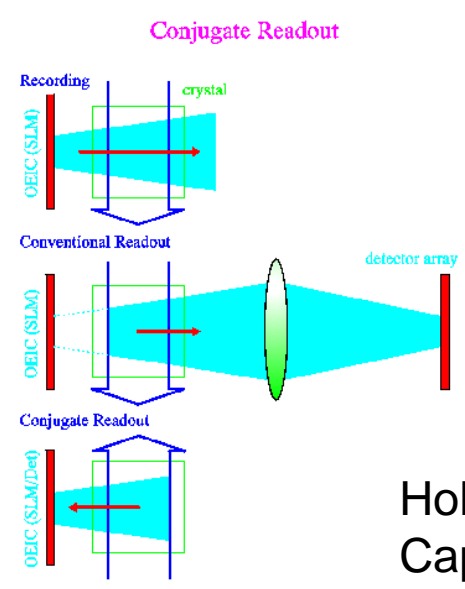
Processor in Memory (PIM) High Memory Bandwidth and Low Power



Superconductor RSFQ logic provides X100 Performance



Data Vortex Optical Communication. Very High Bi-section Bandwidth with Low Latency



Holographic Storage High Capacity with Low Power at Moderate speeds

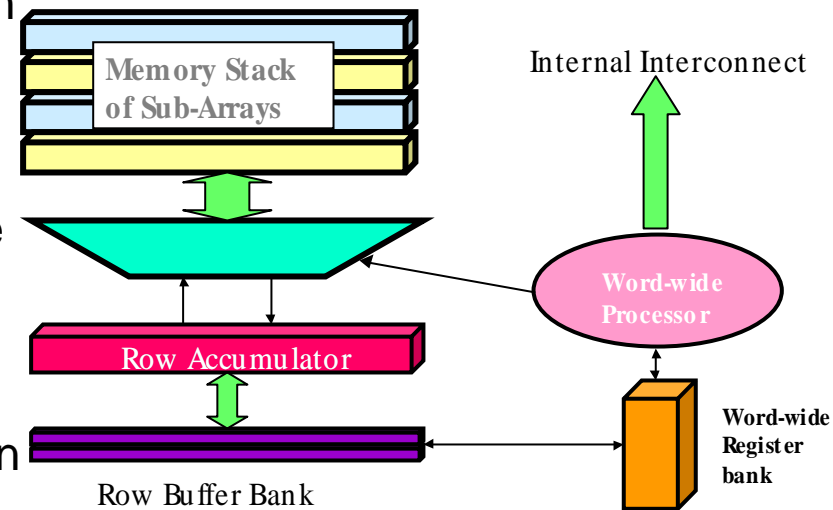
DIVA PIM

Smart Memory for Irregular Data Structures and Dynamic Databases

Processor in Memory

- Merges memory & logic on single chip
- Exploits high internal memory bandwidth
- Enables row-wide in-place memory operations
- Reduces memory access latencies
- Significant power reduction
- Efficient fine grain parallel processing

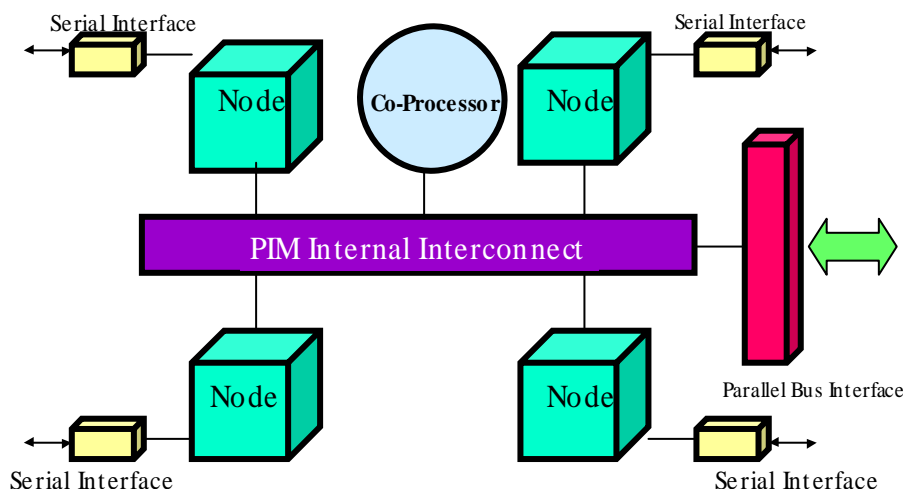
DIVA PIM Node Block Diagram



DIVA PIM Project

- DARPA sponsored \$12.2M USC ISI prime with Caltech (\$2.4M over 4 years), Notre Dame, U of Del
- Greatly accelerate scientific computing of irregular data structures and commercial dynamic databases
- 0.25 μm 256 Mbit part delivered 4Qtr 00
- 4 processor/memory nodes
- **Key innovation of Multithreaded execution for high efficiency through latency management**
- Active message driven object oriented computation
- Direct PIM to PIM interaction without host processor intervention

Major Subsystems



Summary of HTMT

- processor: 150 GHz, 600 Gflops
- # processors: 2048
- memory: 16 Tbytes PIM-DRAM, 80ns access time
- interconnect: Data Vortex, 500 Gbps/channel, > 10 Pbps bi-section bw
- 3/2 storage: 1 Pbyte, 10 us access time
- shared memory, 4 level hierarchy
- latency management: multithreaded with percolation

COTS Cluster Breakout Group

David H. Bailey *

James Bieda

Remy Evard

Robert Clay

Al Geist

Carl Kesselman

David E. Keyes

Andrew Lunsdaine

James R. McGraw

Piyush Mehrotra

Daniel Savarese

Bob Voigt

Michael S. Warren



COTS Cluster Group Findings

Working system model:

- 12,500 nodes.
- Individual nodes: 8-proc. shared memory COTS system.
- Individual processors: 3 GHz, 10 Gflop/s, possibly a multi-CPU design.
- Main memory bandwidth per processor: 10 Mbyte/s.
- Main memory: 20 Gbyte memory per node, 250 Tbyte total.
- Disk space: 200 Gbyte per node, 2.5 Pbyte total.
- Archival storage provided on each node.

Interconnection Network

- 4-deep hierarchy of 16x16 crossbar switches, with more switches at higher levels.
- Optical WDM fiber links, 20 Gbyte/s bandwidth per channel.
- Direct node-to-node communication -- no transmittal through other nodes.
- Thousands of extra ports in the network could be used for external I/O.
- Node-to-node latency: 1 us hardware, 11 us with software.
- Global synchronization or reduction time: 300 us with software.

Front End System

Multi-node front end system with full OS, compilers, etc.

- Allocates nodes, initiates and terminates jobs.
- Assumes only one user job per node -- i.e. space sharing.
- Manages batch queues and system resources.
- Monitors and manages the network and cluster nodes.
- Provides cluster configuration information to nodes.
- Designates nodes up or down.
- Maps spare nodes to replace failed ones.
- Updates runtime software on cluster nodes.
- Provides user interface to cluster nodes for running parallel commands, compiling, debugging, loading input data, handling output data.

Cluster Node System Software

- Stripped-down Linux (or the equivalent).
- Initiates and terminates the application on the node.
- Handles local disk and external network I/O.
- Supports virtual memory.
- Supports performance monitoring.
- Supports shared memory parallel execution.
- Supports fault detection and mitigation.

Programmer's Responsibilities

- Expose enormous levels of parallelism and achieve high data locality.
- Explicitly handle parallelism, tasks, decomposing data, etc.
- Programming model:
 - Within nodes: message passing (i.e. MPI) or shared memory threads (i.e. OpenMP).
 - Between nodes: message passing (i.e. MPI).
- Checkpointing -- user will periodically write required restart data to local disk.

Critical System Software

- A cluster node Unix-based OS (i.e. Linux or the like), scalable to 12,500+ nodes.
- Fortran-90, C and C++ compilers, generating maximum performance object code, usable under the Linux OS.
- An efficient implementation of MPI, scalable to 12,500+ nodes.
- System management and job management tools, usable for systems of this size.

System Software Research Tasks

- Can a stripped down Linux-like operating system be designed that is scalable to 12,500+ nodes?
- Can vendor compilers be utilized in a Linux node environment?
- If not, can high-performance Linux-compatible compilers be produced by third party vendors, keyed to needs of scientific computing?
- Can MPI be scaled to 12,500+ nodes?
- Can system management and batch submission (i.e. PBS or LSF) tools be scaled to 12,500+ nodes?
- Can an effective performance management tool be produced for systems with 12,500+ nodes?
- Can an effective debugger be produced for systems with 12,500+ nodes? Can the debugger being specified by the Parallel Tools consortium be adapted for these systems?

Algorithm/Application Research Tasks

- Is there a sufficiently large class of petaflops-class applications that possess
 - 10^7 to 10^8 concurrency at virtually all steps of the computation?
 - 95+% cache locality and 99+% node locality?
- Are there hierarchical and latency tolerant variants of key algorithms?

In general, what are reliable estimates of achieved performance for various petaflops applications on the proposed system?

MPP Breakout Group



Rudolf
Eigenmann
Jose Fortes
David Frye
Kent Koeninger
Vipin Kumar
John May
Paul Messina
Merrell Patrick
Paul Smith
Rick Stevens *
Valerie Taylor
Josep Torrellas
Paul Woodward



Challenges to Pflops MPP

- Global user shared name space
 - sufficient address size, with user id
 - cache consistency model with hardware support
- Global process id
- Virtual memory and processor
 - protection/reliability
 - translation buffer consistency
 - efficient address mapping
- Intrinsic latency tolerance thru advanced caching
- Dynamic load balancing
- Leverage industry investment in COTS devices

MPP Strategy

- Custom design of performance critical system architecture elements
 - high bandwidth/low latency interconnect network
 - efficient mechanisms for cache management & global coordination
- Exploit COTS where ever possible
- Devise hierarchical system structures for variable latency and graceful locality degradation
- Dynamic resource management
- Keep costs to near cluster levels
- Automatic parallelization/data partitioning/task allocation

MPP Train-Wreck

- Objectives are in conflict
 - global latency tolerance not supported by COTS processors
 - TLB consistency management does not scale
 - Cost advantage of COTS lost to custom design requirements
 - User management of locality still likely
- Language limitations
 - Message-passing imposes parallelism granularity barrier
 - Data structure driven parallelism unavailable due to overhead of interpreting
- System-wide operating system scalability model elusive
- Fine-grain runtime system a research topic
- Conventional cache policies in conflict with memory access patterns of many applications

Recommendations

- Begin scalable O/S design now
- Detailed applications scaling studies essential to determining global requirements
- Some innovative technologies crucial to meeting capacity requirements
- New execution paradigm/programming model
 - experimental API, compiler, runtime based on new metrics
- PIM may be essential for memory bandwidth
- Percolation or other pro-active latency management
- Establish meaningful real-world metrics

HTMT Breakout Group



Larry Bergman *
Nikos Chrisochoides
Vincent Freeh
Guang R. Gao
Peter Kogge
Phil Merkey
John Van Rosendale
John Salmon
Burton Smith
Thomas Sterling



HTMT Discussion Topics

- Synchronization that the runtime handles
- Scheduling (DRAM/SRAM, SRAM/CNET)
- Runtime to thread percolation interface
- API: How should users view machine?
- Performance instrumentation & debug tools

Multilevel Multithreaded Execution Model

- Extend latency hiding of multithreading
- Hierarchy of logical threads
 - Delineates threads and thread ensembles
 - Action sequences, state, and precedence constraints
- Fine grain single cycle thread switching
 - Processor level, hides pipeline and time of flight latency
- Coarse grain context “percolation”
 - Memory level, in memory synchronization
 - Ready contexts move toward processors, pending contexts towards big memory

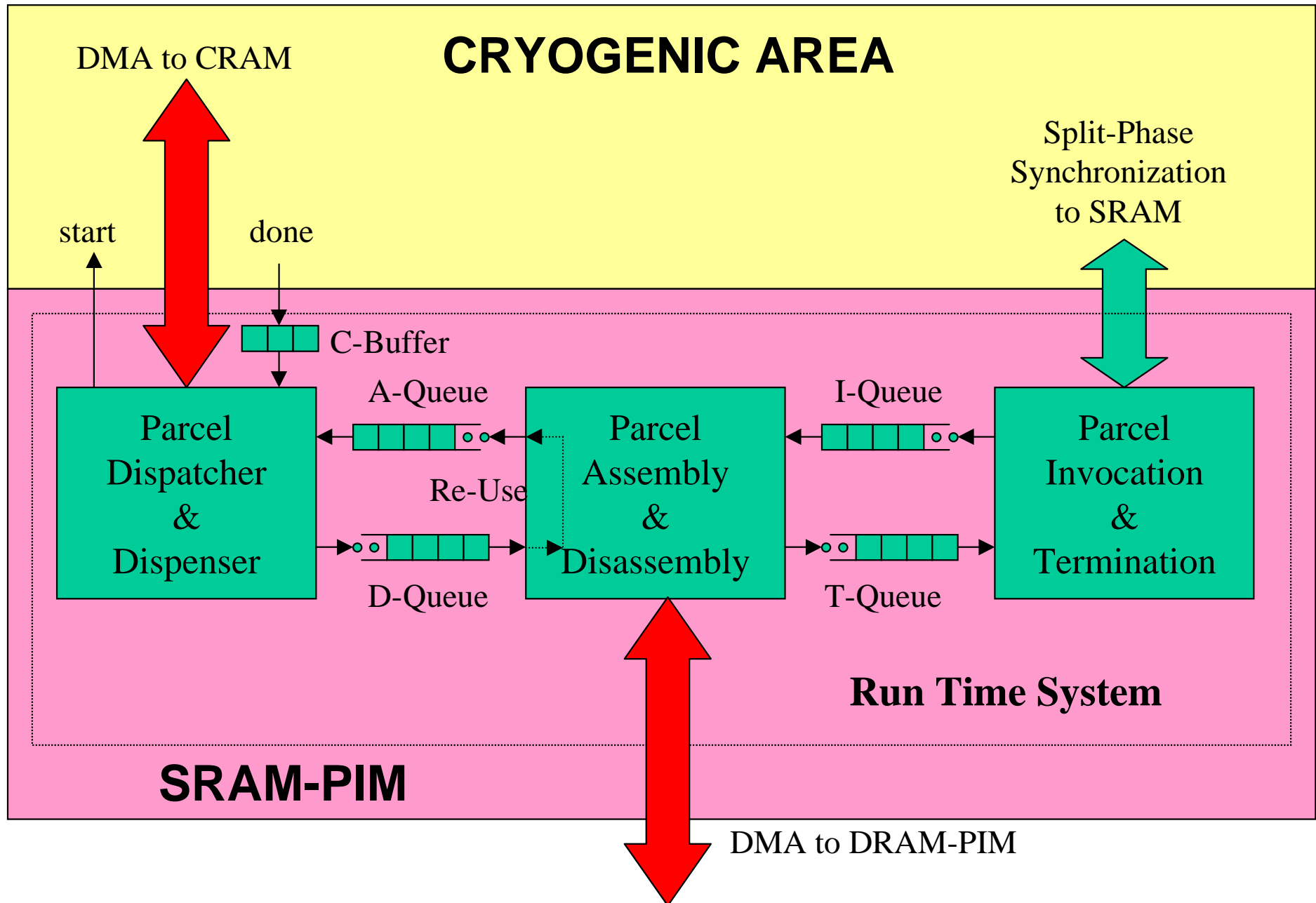
Performance Path

- Identify distinct executable/components entities within HTMT
 - DRAM, SRAM (active, persistent objects)
 - SRAM/DRAM (run time)
 - SPELL executable code
- SPELL Program - to - SRAM Object/Runtime interaction (and SRAM to SRAM) semantics
 - sync ops
 - prefix ops, reduction ops
 - context triggers
 - MALLOC/FREE
- SRAM Runtime - to - SPELL
 - Enq/Deq contexts
 - exchange jumps
 - block context movements
- SRAM - DRAM data permutations
- DRAM active objects execution
 - file objects
 - searches / traversals / de-references
 - compression/decompression operations
 - data intensive regular ops

Decision Path

- How does compiler/programmer/runtime decide:
 - What runs?
 - Where does it run?
 - What data is needed to allow it to run?
 - When is data needed, and where is it?
 - How is data reused?
 - What mechanisms are invoked (or policy) to select/dispatch it?
 - How do we “chain” between different runnable units?
- What instrumentation is activated to debug/optimize?

HTMT Percolation Model



Unresolved Issues

- A language notation to explicitly reflect the different levels of operations
- Is this a multi-programmed system?
- What is the minimum compiler support?
 - E.g., loop distribution for software pipelining
- How do external computers communicate with the HTMT file system?
 - How to get data on? How to load programs?

Unresolved Issues (2)

- Scheduling policies
 - user directed, runtime, compiler, ...
- Fault management
- API: How should users view machine?
 - What should programmers worry about?
 - What is tradeoff space?
- Performance model, instrumentation & debug tools

Task Definitions

- Build the runtime
- Fault discovery and reconfiguration
- Determine critical memory reuse factors
- Develop user level resource management mechanisms
- Define and implement data motion library
- Develop the file system concept
- Develop a straw man HTMT programming manual/guide
- Develop performance instrumentation strategy
 - metrics, probe methods, tradeoffs, analysis methods

Summary Findings

- Architecture is important
 - Bandwidth requirements dominate hardware structures
 - Latency management determines runtime resource management strategy
 - Efficient mechanisms for overhead services
- Generality of application workload dependent on interconnect throughput and response time
- COTS processors will not hide system latency, even if multithreading is adopted
- More memory than earlier thought may be needed
- MPP problem is very difficult, unclear which direction to take

Summary Findings (cont)

- COTS clusters will provide safe migration path at best price-performance but must rely on user management of all system resources
- Inter-process load balancing too expensive on clusters
- New formalism required to expose diverse modes of parallelism
- Compilers can't ever make all performance decisions; must be combined with collaborative runtime software
- Critical-path performance decision tree requires new internal protocols
- User must describe application properties, not means

Open Issues

- Is network of processor/memories best use of multi-billion transistor chips
- Is convergence real or only point of inflexion
- Will semiconductor continue to push beyond 0.15 micron; do market costs support it
- Can alternative technology fabrication be supported/avoided
- Can orders-of-magnitude latency be managed
- What will the computer languages of the Pflops era look like?
- Processor granularity: fine and many or fat and few

Major Recommendations

- PLEASE! No more workshops
- Sponsor point-design studies for system software models for MPP and COTS Clustered approaches
 - elaborate on functionality of performance critical elements
 - establish decision path and interrelationships
- Implement low level HTMT emulator on dedicated Beowulf-class platform
 - modeled functional elements map isomorphically onto physical computing elements and communications channels
 - evaluate percolation and PIM gather-scatter
- HTMT requires phase 3 funding FY00
- Derive definitive CMOS roadmap

Major Recommendations

- Select 2 or 3 applications that
 - need petaflops
 - have diverse algorithms
 - have different memory size and access patterns
- Derive configurations for the MPP and exotic architectures that would support those systems
- Assess programmability, likely fraction of peak speed that would be achievable
- Sponsor development of stripped down version of Linux for MPP and Clustered approach that is robust and scalable to tens of thousands of nodes
- Reliability must become intrinsic to applications software